**MammothDB Engine**

Supported SQL as of September 2015



1. Supported Data Types 3

1.1. Numeric Data Types 3

1.2. Date and Time Data Types 4

1.3. String Data Types 4

1.4. Not Supported Data Types 4

2. Data Definition Statements 6

2.1. Create Database 6

2.2. Create Table 6

2.3. Drop Database 6

2.4. Drop Table 6

2.5. Use Database 6

2.6. Truncate Table 7

2.7. Alter Database 7

2.8. Alter Table 7

2.9. Rename Table 7

3. Data Manipulation Statements 8

3.1. Delete 8

3.2. Insert 8

3.3. Update 8

3.4. Load Data 8

3.5. Select 8

3.6. Replace 9

Distinct & Group by Limitations 10

3.7. Distinct 10

3.8. Group by 10

4. Joins 11

4.1. Inner Join 11

4.2. Left Outer Join 11

4.3. Right Outer Join 11

4.4. Cross Join 12

4.5. Natural Join 12

5. Functions and Operations 13

5.1. Aggregate Functions 13

5.2. Mathematical Functions 13

5.3. Comparison Functions and Operators 14

5.4. String Functions 15

5.5. Date and Time Functions 16

5.6. Control Flow Functions 18

# Supported Data Types

Data types that are supported in MammothDB are based on data types used in MySQL.

This article provides an overview on these data types with the restrictions imposed by MammothDB implementation.

Data type descriptions use these conventions:

* M or (length) indicates the maximum display width for integer types. For floating-point and fixed-point types, M is the total number of digits that can be stored (the precision). For string types, M is the maximum length. The maximum permissible value of M depends on the data type.
* D applies to floating-point and fixed-point types and indicates the number of digits following the decimal point (the scale).
* Square brackets (“[” and “]”) indicate optional parts of type definitions.

For more details about usage and meaning of the described data types see official MySQL Reference Manual.

## Numeric Data Types

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Supported Status** | **Range** |
| TINYINT [(length)] | Fully Supported | From -127 to 127 |
| BOOL | Fully Supported | From -127 to 128 |
| BOOLEAN | Fully Supported | From -127 to 129 |
| SMALLINT [(length)] | Fully Supported | From -32767 to 32767 |
| MEDIUMINT [(length)] | Fully Supported | From -8388608 to 8388608 |
| INT [(length)] | Fully Supported | From -2147483647 to 2147483647 |
| BIGINT [(length)] | Fully Supported | From -9223372036854775806 to 9223372036854775806 |
| FLOAT [(length,decimals)] | Fully Supported | From -3.402823466E+38 to 3.402823466E+38 |
| DOUBLE [(length,decimals)] | With MDB Restriction | From -1.7976931348623157E+307 to 1.7976931348623157E+307 |
| DECIMAL(M,D) | With MDB Restriction | 1 <= M <= 18, 0 <= D < M |

## Date and Time Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **Supported Status** | **Range** | **Definition Constraint** |
| DATE | Fully Supported | 1000-01-01 to 9999-12-31 |  |
| TIMESTAMP | With Infobright Restriction | 1970-01-01 00:00:01 to 2038-01-01 00:00:00 | col\_name timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP |
| TIME | With MDB Restriction | 00:00:00 to 23:59:59 |  |
| DATETIME | Fully Supported | 1000-01-01 00:00:00 to 9999-12-31 23:59:59 |  |

## String Data Types

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **Supported Status** | **Range** | **Definition Constraint** |
| CHAR(M) | With MDB Restriction | 1 <= M <= 255 for latin1 charset;  1 <= M <= 85 for UTF8 charset | Boundary values for M must be used carefully, with accordance of defined in MySQL row size constraints |
| VARCHAR(M) | With MDB Restriction | 1 <= M <= 21844 | Boundary values for M must be used carefully, with accordance of defined in MySQL row size constraints |

## Not Supported Data Types

BIT (synonym for TINYINT(1))

INTEGER (synonym for INT)

SERIAL (an alias for BIGINT UNSIGNED NOT NULL AUTO\_INCREMENT UNIQUE)

DECIMAL(M,D) (for M > 18)

YEAR[(2|4)]

CHARACTER SET

CHAR BYTE (an alias for the BINARY)

CHAR(0)

BINARY

VARBINARY

TINYBLOB

TINYTEXT

TEXT

MEDIUMTEXT

LONGTEXT

BLOB

MEDIUMBLOB

LONGBLOB

# Data Definition Statements

Data definition statements supported by MammothDB do not differ much from standard ANSI SQL syntax. The differences imposed by MammothDB design are mainly in CREATE TABLE syntax.

The engine must be defined as MDB during creation of the table. In MammothDB versions up to 0.32.1 it is needed to use D or F prefix for table name in order engine to recognize the type of the table. F prefix must be used for creation of partitioned tables and D prefix – for table replicated on each cluster node.

## Create Database

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db\_name

[  [DEFAULT] CHARACTER SET [=] charset\_name | [DEFAULT] COLLATE [=] collation\_name]

*Note:* Create database does not differ from MySQL implementation

## Create Table

CREATE TABLE [IF NOT EXISTS] tbl\_name

(col\_name data\_type [NOT NULL | NULL] [DEFAULT default\_value],

...)

ENGINE=MDB DEFAULT CHARSET=UTF8 [PARTITION BY KEY (col\_name)]

or

CREATE TABLE [IF NOT EXISTS] tbl\_name

ENGINE=MDB [DEFAULT CHARSET=UTF8] select\_statement

or

CREATE TABLE [IF NOT EXISTS] tbl\_name { LIKE old\_tbl\_name | (LIKE old\_tbl\_name) }

*Note:* tbl\_name: Table name must start with F or D in order to be recognized as partitioned or replicated.

data\_type:  for supported data types see [Data Types](http://alfresco.xentio.local:8080/share/page/site/mammothdb/wiki-page?title=Data_types&listViewLinkBack=true)

## Drop Database

DROP DATABASE  [IF EXISTS] db\_name

*Note:* Drop schema is not supported

## Drop Table

DROP TABLE [IF EXISTS] tbl\_name [, tbl\_name] …

## Use Database

USE db\_name

## Truncate Table

TRUNCATE [TABLE] tbl\_name

## Alter Database

ALTER {DATABASE | SCHEMA} [db\_name][DEFAULT] CHARACTER SET [=] charset\_name| [DEFAULT] COLLATE [=] collation\_name

## Alter Table

Not supported

## Rename Table

Not supported

# Data Manipulation Statements

## Delete

Not supported

## Insert

INSERT [INTO] tbl\_name

[(col\_name,...)]

{VALUES | VALUE}

({expr | DEFAULT},...),

(...),

…

;

or

INSERT [INTO] tbl\_name

SET col\_name={expr | DEFAULT},

col\_name=…

;

or

INSERT [INTO] tbl\_name [(col\_name,...)

SELECT … ;

## Update

Not supported

## Load Data

LOAD [DATA [LOCAL] INFILE 'file\_name' INTO TABLE tbl\_name [CHARACTER SET charset\_name]

[{FIELDS | COLUMNS}[TERMINATED BY 'string']

[ [OPTIONALLY] ENCLOSED BY 'char']

  [ESCAPED BY 'char'] ]

  [LINES[STARTING BY 'string']

  [TERMINATED BY 'string']

]

[IGNORE number LINES];

## Select

SELECT [ALL | DISTINCT ] select\_expr [, select\_expr ...]

[FROM table\_references

[WHERE where\_condition]

[GROUP BY {col\_name | expr}[ASC | DESC], ... ]

[HAVING where\_condition]

[ORDER BY {col\_name | expr }[ASC | DESC], ...]

[LIMIT {[offset,] row\_count}]

[INTO OUTFILE 'file\_name' export\_options]

];

*Note*: order by {position} and group by {position} are not supported

## Replace

This is MySQL extension and it is not supported. Command doesn't display error and works like pure insert.

Distinct & Group by Limitations

## Distinct

SELECT DISTINCT  col\_name, col\_name, ...

FROM table\_references

[WHERE where\_condition]

[ORDER BY {col\_name | expr | position}[ASC | DESC], ...][LIMIT {[offset,] row\_count}]

[INTO OUTFILE 'file\_name' export\_options]

or

SELECT DISTINCT group\_col\_name\_list

FROM table\_references

[WHERE where\_condition]

GROUP BY group\_col\_name\_list [HAVING where\_condition]

[ORDER BY col\_name

[ASC | DESC], ...][LIMIT {[offset,] row\_count}]

[INTO OUTFILE 'file\_name' export\_options]

or

SELECT group\_col\_name\_list, AGGR(DISTINCT col\_name)

FROM table\_references

[WHERE where\_condition]

GROUP BY group\_col\_name\_list [HAVING where\_condition]

[ORDER BY col\_name [ASC | DESC], ...]

[LIMIT {[offset,] row\_count}]

[INTO OUTFILE 'file\_name' export\_options]

*Note: Only one AGGR function over distinct values of one column is supported. See*[*list with supported aggregation functions*](http://alfresco.xentio.local:8080/share/page/site/mammothdb/wiki-page?title=Functions_and_Operations&listViewLinkBack=true#Aggregate_Functions) *for more datails.*

## Group by

SELECT group\_col\_name\_list, list\_AGGR(col\_name)

FROM table\_references

[WHERE where\_condition]

GROUP BY group\_col\_name\_list [HAVING where\_condition]

[ORDER BY col\_name [ASC | DESC], ...]

[LIMIT {[offset,] row\_count}]

[INTO OUTFILE 'file\_name' export\_options]

*Note: Only columns listed in group clause can be selected together with aggregation functions over any of the table column (as per ANSI SQL specification).*[*MySQL extension for GROUP BY*](http://dev.mysql.com/doc/refman/5.6/en/group-by-handling.html) *is not supported.*

# Joins

## Inner Join

SELECT select\_expr [, select\_expr ...]

FROM

tbl\_name\_1,

tbl\_name\_2,

…,

tbl\_name\_n

WHERE

tbl\_name\_1.id\_col = tbl\_name\_2.id\_col and

tbl\_name\_2.id\_col = tbl\_name\_3.id\_col and

…

or

SELECT select\_expr [, select\_expr ...]

FROM tbl\_name\_1

[INNER] JOIN tbl\_name\_2

ON tbl\_name\_1.id\_col = tbl\_name\_2.id\_col;

## Left Outer Join

SELECT select\_expr [, select\_expr ...]

FROM tbl\_name\_1

LEFT [OUTER] JOIN tbl\_name\_2

ON tbl\_name\_1.id\_col = tbl\_name\_2.id\_col;

*Note:*

*MDB supports LEFT JOIN between partitioned in left and replicated tables in right and between replicated tables.*

I.e. F\_tbl with D\_tbl or D\_tbl with D\_tbl.

*The case D\_tbl left joined to F\_tbl returns more rows than expected.*

## Right Outer Join

SELECT select\_expr [, select\_expr ...]

FROM tbl\_name\_1

RIGHT [OUTER] JOIN tbl\_name\_2

ON tbl\_name\_1.id\_col = tbl\_name\_2.id\_col;

*Note:*

*MDB supports RIGHT JOIN between replicated in left and partitioned table in right and between replicated tables.*I.e. D\_tbl with F\_tbl or D\_tbl with D\_tbl.*The case F\_tbl right joined to D\_tbl returns more rows than expected.*

## Cross Join

SELECT select\_expr [, select\_expr ...]

FROM tbl\_name\_1

CROSS JOIN tbl\_name\_2;

*Note: MDB supports standard SQL CROSS JOIN definition.*

## Natural Join

SELECT select\_expr [, select\_expr ...]

FROM tbl\_name\_1 NATURAL JOIN tbl\_name\_2;

*Note: Deviation from MySQL natural join definition:*

*Columns with same name of associate tables will appear as many times as the number of tables*

# Functions and Operations

This article describes the functions and operations used in MySQL and known to be supported by MammothDB. The detailed description of the usage is not provided as it can be found in official [MySQL Reference Manual](http://dev.mysql.com/doc/refman/5.1/en/index.html). Here the exact syntax supported in MDBE is described and the differences and possible implications are also highlighted.

## Aggregate Functions

**Count**

COUNT(\*),

COUNT([DISTINCT] column\_name)

**Avg**

AVG([DISTINCT] X)

*Note:X is a numeric data type or expression*

**Sum**

SUM([DISTINCT] X)

*Note: X is a numeric data type or expression*

**Max**

MAX([DISTINCT] X)

*Note: X is a numeric data type, DATE or expression.*

*MAX over the string returns undefined result*

**Min**

MIN([DISTINCT] X)

*Note:X is a numeric or date-time data type or expression.*

*MIN over the string returns undefined result*

## Mathematical Functions

**Round**

ROUND(X)

ROUND(X,D)

*Note: X is a numeric data type*

**Sqrt**

SQRT(X)

*Note:X is a non negative number*

**Abs**

ABS(X)

*Note: X is a numeric data type or expression*

**Ceiling**

CEILING(X)

or

CEIL(X)

*Note: X is a numeric data type or expression*

**Power**

POW(X,Y)

or

POWER(X,Y)

*Note: X is a numeric data type or expression*

**Sign**

SIGN(X)

*Note: X is a numeric data type or expression*

**Truncate**

TRUNCATE(X,D)

**Floor**

FLOOR(X)

*Note: X is a numeric data type or expression*

**Bitwise operations**

& - Bitwise AND;

| -  Bitwise OR;

^ -  Bitwise XOR

**Arithmetic operator**

'+, -, \*, /, DIV, %, MOD

*Note: Operands are applicable for numeric data types*

## Comparison Functions and Operators

**Coalesce**

COALESCE(value,...)

*Note: MDBE doesn't support usage of column alias as value for COALESCE.*

**Between**

expr BETWEEN min AND max

**Comparison Operators**

 =

>

<

>=

<=

<>

!=

**NULL-safe Еqual**

<=>

**Checks Whether Value Is NULL**

IS NULL

**Checks Whether Value Is Not NULL**

IS NOT NULL

**In List**

expr IN (value,...)

**Not In List**

expr NOT IN (value,...)

**Function ISNULL**

ISNULL(expr)

## String Functions

**Char\_length**

CHAR\_LENGTH(str)

CHARACTER\_LENGTH(str)

**Substring**

SUBSTRING(str,pos),

SUBSTRING(str FROM pos),

SUBSTRING(str,pos,len),

SUBSTRING(str FROM pos FOR len)

*Note: SUBSTR() is a synonym for SUBSTRING().*

**Length**

LENGTH(str)

**Left**

LEFT(str,len)

**Trim**

TRIM(str)

*Note: The rest of the full TRIM functionality defined in MySQL is not supported in MDBE*

**Reverse**

REVERSE(str)

**Right**

RIGHT(str,len)

**Lower**

LOWER(str)

**Locate**

LOCATE(substr,str)

LOCATE(substr,str,pos)

**Pattern Matching LIKE**

expr LIKE pat

**Not Matching To Pattern**

expr NOT LIKE pat

**Cstring Concatenation**

CONCAT(str1,str2,...)

**String To Date**

STR\_TO\_DATE(str,format)

**Format**

FORMAT(X,D)

*Note: X is a numeric data type or expression. Returns string data type.*

## Date and Time Functions

**Add Time Values (Intervals) To A Date Value**

ADDDATE(expr,days)

*Note:* ADDDATE(date,INTERVAL expr unit) *is not supported correctly and usage may produce wrong results*

**Add Time**

ADDTIME(expr1,expr2)

*Note: expr1 is DATETIME value, expr2 is a TIME value.*

*MDBE doesn't support this function for expr1=TIME value*

**Convert Time zone**

CONVERT\_TZ(dt,from\_tz,to\_tz)

*Note: see official MySQL reference guide for details*

**Return The Current Date**

CURDATE(),

CURRENT\_DATE,

CURRENT\_DATE()

*Note: MySQL layer. Function and its synonyms are not processed by MDBE*

**Returns The Current Time**

CURTIME(),

CURRENT\_TIME,

CURRENT\_TIME()

*Note: MySQL layer. Function and its synonyms are not processed by MDBE*

**Returns The Current Date and Time**

NOW(),

CURRENT\_TIMESTAMP,

CURRENT\_TIMESTAMP()

*Note: MySQL layer. Function and its synonyms are not processed by MDBE*

**Extracts The Date Part**

DATE(expr)

*Note: expr is DATE or DATETIME value*

**A Value In Days From One Date to the Other**

DATEDIFF(expr1,expr2)

*Note: expr1 and expr2 are date or date-and-time expressions]*

**Add Time Values (Intervals) to a Date Value**

DATE\_ADD

*Note: NOT SUPPORTED use ADD\_DATE instead*

Instead, use the following syntax:

date + INTERVAL expr unit

*Note: Same date arithmetic as DATE\_ADD.*

*See official MySQL reference guide for usage details*

**Subtract Time Values from a Date Value**

DATE\_SUB

*Note: NOT SUPPORTED*

Instead, use the following syntax:

date - INTERVAL expr unit

*Note: Same date arithmetic as DATE\_SUB.*

*See official MySQL reference guide for usage details*

**Format Date Value**

DATE\_FORMAT(date,format)

*Note: See official MySQL reference guide for usage details*

**Return Day From Date**

DAYOFMONTH(date)

DAY(date)

*Note: Both are synonyms*

**Returns The Weekday Index for Date**

 DAYOFWEEK(date)

**Returns The Day Number in The Year**

DAYOFYEAR(date)

*Note: EXTRACT(unit FROM date) is NOT SUPPORTED*

**Returns A Date Format String**

GET\_FORMAT({DATE|TIME|DATETIME}, {'EUR'|'USA'|'JIS'|'ISO'|'INTERNAL'})

**Extract the hour**

HOUR(time)

**Return the last day of the month for the argument**

LAST\_DAY(date)

**Return the minute from the argument**

MINUTE(time)

**Return the month from the date passed**

MONTH(date)

**Return the name of the month**

MONTHNAME(date)

**Return the quarter from a date argument**

QUARTER(date)

**Return the second (0-59)**

SECOND(time)

## Control Flow Functions

**IF**

IF(expr1,expr2,expr3)

*Note: If expr1 is TRUE (expr1 <> 0 and expr1 <> NULL) then IF() returns expr2; otherwise it returns expr3.*

**CASE**

CASE WHEN [condition] THEN result [WHEN [condition] THEN result ...] [ELSE result] END

**IFNULL**

IFNULL(expr1,expr2)

*Note:* If expr1 is not NULL, IFNULL() returns expr1; otherwise it returns expr2. IFNULL() returns a numeric or string value, depending on the context in which it is used.

**NULLIF**

NULLIF(expr1,expr2)

*Note: Returns NULL if expr1 = expr2 is true, otherwise returns expr1.*